

Teoría de Lenguajes
Preguntas de examen final
17 noviembre 2022

Ejercicio 1. Dar un algoritmo que decida si dos expresiones regulares denotan el mismo lenguaje. Justificar la correctitud. Analizar la complejidad computacional de peor caso.

Ejercicio 2. Demostrar que dada una gramática libre de contexto G , hay existe una constante c tal que para todo par de símbolos no terminales A, B , para toda cadena de terminales w y para toda cadena de terminales y no terminales α ,

$$\text{si } A \xRightarrow[i]{L} wB\alpha, \text{ entonces } i \leq c^{|w|+2}.$$

Ejercicio 3. Dar dos algoritmos distintos para determinar si el lenguaje aceptado por un autómata finito dado es el conjunto de todas las cadenas del alfabeto. Justificar cada uno y dar su complejidad algorítmica.

Ejercicio 4. Dar un algoritmo que determine si un lenguaje regular dado es infinito. Justificar y dar la complejidad del algoritmo en el peor caso.

Ejercicio 5. ¿Cuántos autómatas finitos deterministas con dos estados pueden construirse sobre el alfabeto $\{0, 1\}$?

¿Cuántos autómatas finitos no determinísticos con dos estados pueden construirse sobre el alfabeto $\{0, 1\}$?

¿Cuántos autómatas de pila con dos estados pueden construirse con alfabeto de entrada A , alfabeto de pila Z , y una cantidad máxima M símbolos en cada transición?

Ejercicio 6. Fijados los alfabetos Δ y Γ , ¿Cuántos autómatas de pila distintos $(Q, \Sigma, \Delta, \Gamma, q_0, F)$ determinísticos hay, Si Q tiene 5 estados y en cada transición se escriben en la pila 0, 1 o 2 símbolos? ¿Y cuántos no determinísticos?

Ejercicio 7. Dar la definición de gramática libre de contexto recursiva a derecha. Dar un ejemplo. Dar el algoritmo de eliminación de recursión a derecha (inmediata y no inmediata), su justificación de correctitud, y su complejidad computacional.

Ejercicio 8. Dar el algoritmo de pasar a forma normal 3-chomsky. Justificar correctitud y dar la complejidad computacional.

$$A \rightarrow a$$

$$A \rightarrow BC$$

$$A \rightarrow BCD$$

No se permiten producciones $A \rightarrow B$.

donde A, B, C, D son no terminales y a es terminal.

Entonces son 3-Chomsky

$$S \rightarrow ABC$$

$$A \rightarrow BDE$$

$$A \rightarrow a$$

$$A \rightarrow BC$$

No son 3-Chomsky

$$A \rightarrow B$$

$A \rightarrow ABCDE$ tampoco es 3-Chmsky

$A \rightarrow abcdef$ tampoco es 3-Chomsky

Ejercicio 9. Consideremos el transductor finito dado por una máquina de Mealy $(S, S_0, \Sigma, \Gamma, T, G)$

,

S es un conjunto finito de estados

S_0 es un estado inicial

Σ es el alfabeto de entrada

Γ es el alfabeto de salida

$\delta : S \times \Sigma \rightarrow S$ es la función de transición

$\gamma : S \times \Sigma \rightarrow \Gamma$ mapea un estado y un símbolo de entrada a un símbolo de salida

Adaptar el algoritmo de minimización de autómatas finitos a una minimización de máquina Mealy.

Ayuda: Definir la relación de equivalencia considerando la función δ extendida y la función gamma extendida.

Ejercicio 10.

a) Demostrar lo siguiente: Si un autómata finito es determinístico, accesible, coaccesible y codeterminista entonces es mínimo.

accesible: Todos los estados son accesibles desde el estado inicial.

co-accesible: Todos los estados tienen un camino en el autómata hasta un estado final

co-determinístico: hay un único estado final y no hay transiciones $\delta(p, a) = q$ y $\delta(r, a) = q$,

b) Demostrar que la recíproca de la afirmación en a) no siempre es cierta.

Ejercicio 11. Demostrar que dada una gramática regular a derecha se puede obtener una gramática regular a izquierda equivalente. Tener en cuenta que disponemos del algoritmo para ir de gramática regular a derecha a autómata finito y que también disponemos del algoritmo para ir de autómata finito a gramática regular a derecha.

Ayuda: hallar la gramática del reverso de un lenguaje y el autómata finito del reverso de un lenguaje, o sea, dada G tal que $L = L(G)$ hallar GR tal que $LR = L(GR)$ y dado M tal que $L = L(M)$ hallar MR tal que $LR = L(MR)$, donde LR es el reverso del lenguaje L .

Ayuda adicional: Hacer un ejemplo de gramática con 2 no terminales y 2 terminales y que genere una sola cadena.

Posible ayuda adicional: invertir producciones y transiciones.

Ejercicio 12. Dado un autómata finito no determinístico A pero sin transiciones lambda dar un algoritmo que construye el autómata finito no determinístico que acepta el lenguaje $L(A)L(A)^R$. Demostrar por inducción en el largo de las cadenas de que el algoritmo es correcto. Determinar la complejidad computacional del algoritmo.

Ejercicio 13. Definimos un autómata de pila de doble entrada $P = (Q, \Sigma, \tau, \Gamma, \delta, q_0, Z_0, F)$ donde

$$\delta : Q \times \Sigma \cup \{\lambda\} \times \tau \cup \lambda \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma^*).$$

La función δ es tal que las transiciones lambda ocurren en ambas cintas a la vez. Es decir, no hay transiciones que lean de una cinta y no de la otra.

Demostrar que para todo automata de pila de doble entrada que acepta por estado final siempre se puede encontrar otro equivalente que acepta por pila vacía.

Ejercicio 14. Dado un automata de pila P determinístico dar el autómata de pila no determinístico que acepta $L' = \{wv^R : w, v \in L(P)\}$ donde w^R es la reversa de w , es decir si $w = abcw^R = cba$.

Ejercicio 15. Mostrar que si L es un lenguaje aceptado por un autómata de pila determinístico por pila vacía entonces ninguna palabra de L es prefijo propio de otra palabra de L . ¿Vale lo mismo en caso de que el autómata de pila sea no-determinístico?

Ejercicio 16. Dado R un lenguaje regular, y dado L un lenguaje libre de contexto determinístico, ¿Es decidible si $L=R$? Es decir, ¿hay un algoritmo capaz de decidir la igualdad? En caso de que sí, dar tal algoritmo y justificar. En caso de que no, dar la demostración de indecidibilidad.

Respuesta: Teorema 10.6 Hopcroft 1976

Ejercicio 17. Dado R un lenguaje Regular, y dado L un lenguaje libre de contexto determinístico, ¿Es decidible si R está incluido en L ?

Respuesta: Teorema 10.6 Hopcroft 1976

Ejercicio 18. Dar un algoritmo que transforma una gramática $LL(k)$ en otra $LL(k)$ de la forma

$$A \rightarrow a\alpha \text{ y } A \rightarrow \lambda.$$

donde a es terminal y A es no terminal. Argumentar por qué todas las producciones admiten esta transformación (la gramática original no es recursiva a izquierda).

Ejercicio 19. Considerar la gramática $S \rightarrow SS|a$

- a) Contar la cantidad de árboles de derivación más a la izquierda de a^n
- b) Contar la cantidad de árboles de derivación más a la derecha de a^n

AYUDA: Ensayar con $n = 1, n = 2, n = 3, n = 4, n = 5$ hasta conseguir la forma general.

Ejercicio 20. Dar un algoritmo que transforme cada gramática libre de contexto G sin producciones $A \rightarrow \lambda$ en otra G' que reconozca el mismo lenguaje pero tal que cada producción es de la forma $A \rightarrow a\alpha$, con a un símbolo terminal y α una cadena de no-terminales (puede ser vacía). Justificar la correctitud y dar la complejidad del algoritmo.

Ayuda: Definir un orden parcial $<$ entre los no terminales de G donde si $A \rightarrow B\beta$ entonces $A < B$. Iterar cambiando la gramática donde cada producción con cabeza A tenga un cuerpo que sea un terminal o con un no-terminal B donde $A < B$.

Otra Ayuda: esta es la forma normal de Greibach.

Ejercicio 21. Dar un algoritmo que transforme cada gramática libre de contexto G sin producciones $A \rightarrow \lambda$ en otra G' que reconozca el mismo lenguaje pero es tal que ninguna producción tiene un lado derecho con dos no-terminales seguidos. Justificar la correctitud y dar la complejidad del algoritmo.

Ayuda: Pasar primero a forma normal de Greibach.

Ejercicio 22. Dar un algoritmo que transforme cada gramática libre de contexto G en otra G' que reconozca el mismo lenguaje pero es tal que si $X_1...X_k$ es el lado derecho de una producción entonces todos los símbolos $X_1, ..., X_k$ son distintos. Justificar la correctitud y dar la complejidad del algoritmo